

Making Request Tracker Do What You Mean

Kevin Falcone

Senior Software Engineer | Patch Rejector | Release Engineer | jbsheet

Prerequisites & Assumptions

- You know RT is a "Ticketing System"
- You have RT installed
 - or have tried
- You have used RT
- You know some perl

RT Layout

- `/opt/rt3`
 - `lib`
 - `share/html`
 - `var`
 - `local`

RT 3.6 Plugins

- Normal perl modules
- perl Makefile.PL
- make initdb
- make install

RT 3.6 Plugins

local/lib/RT/

- |-- FM
 - | |-- Article.pm
 - | |-- ArticleCollection.pm
 - | |-- ArticleCollection_Overlay.pm
 - | |-- Article_Overlay.pm
 - | |-- Class.pm
 - | |-- ClassCollection.pm
 - | |-- ClassCollection_Overlay.pm
 - | |-- Class_Overlay.pm
 - | |-- Introduction.pod
 - | |-- ObjectTopic.pm
 - | |-- ObjectTopicCollection.pm
 - | |-- ObjectTopicCollection_Overlay.pm
 - | |-- Record.pm
 - | |-- SearchBuilder.pm
 - | |-- System.pm
 - | |-- Topic.pm
 - | |-- TopicCollection.pm
 - | |-- TopicCollection_Overlay.pm
 - | `-- Topic_Overlay.pm
- |-- FM.pm
- `-- URI
 - |-- a.pm
 - `-- fsck_com_rtfm.pm

share/html/RTFM/

- |-- Article
 - | |-- Delete.html
 - | |-- Display.html
 - | |-- Edit.html
 - | |-- Elements
 - | |-- EditBasics
 - | |-- EditCustomFields
 - | |-- EditLinks
 - | |-- EditTopics
 - | |-- LinkEntryInstructions
 - | |-- Preformatted
 - | |-- SearchByCustomField
 - | |-- SelectSavedSearches
 - | |-- SelectSearchPrivacy
 - | |-- ShowHistory
 - | |-- ShowLinks
 - | |-- ShowSavedSearches
 - | |-- ShowSearchCriteria
 - | |-- ShowSearchResults
 - | |-- ShowTopics
 - | `-- Tabs
- |-- ExtractFromTicket.html
- |-- ExtractIntoClass.html
- |-- ExtractIntoTopic.html
- |-- History.html
- |-- PreCreate.html
- `-- Search.html

share/html/Callbacks/RTFM/

- |-- Admin
 - | |-- Elements
 - | | |-- CustomFieldTabs
 - | | | `-- Default
 - | | `-- Tabs
 - | | `-- Default
- |-- Global
 - | | `-- CustomFields
 - | | `-- index.html
 - | | `-- Default
- | `-- index.html
- | `-- Default

- |-- Elements
 - | |-- EditLinks
 - | | `-- ExtraLinkInstructions
- |-- Header
 - | | `-- Head
- |-- MessageBox
 - | | `-- Default
- | `-- Tabs
 - | `-- Default

RT 3.6 Plugins

- Difficult to remove / upgrade
- Items aren't logically grouped
- Easy to conflict Overlay .pm files
- You have to remove code to disable an extension

RT 3.8 Plugins

- Same installation
 - `RTHOME=/path/to/rt perl Makefile.pl`
 - `make initdb`
 - `make install`

RT 3.8 Plugins

- Disable/Enable in config
- `Set(@Plugins(qw(Your::Plugin Plugin::Two))));`
- Lives under `local/plugins/Plugin-Name`

RT 3.8 Plugins

```
yapc/3.8/local/plugins$ ls  
RT-FM
```

```
yapc/3.8/local/plugins$ ls RT-FM/  
etc  html  lib  po
```

```
yapc/3.8/local/plugins$ ls RT-FM/html/  
Admin          Callbacks      NoAuth        RTFM          SelfService
```

RT 3.8 Plugin Can'ts

- Need to restart apache to enable/disable
- Can't disable or easily remove db changes
- Can collide in Callback and Overlay namespaces

Ways to override RT

- Start with RT_Config.pm
- HTML::Mason callbacks
- HTML::Mason overlays
- perl overlays

RT_Config.pm

- All the options are documented, look there before you start hacking
- `RT->Config->Get('Name')`
 - `RT->Config->Get('Name',$session{CurrentUser})`
- `RT::Config`

HTML::Mason Callbacks

```
/opt/rt3/share/html/Elements/Login
```

```
$m->callback( %ARGS, CallbackName => 'BeforeForm' );
```

```
<& /Elements/Callback, %ARGS,  
_CallbackName => 'BeforeForm' &>
```

HTML::Mason Callbacks

- `local/html/path/to/element/callbackname`

HTML::Mason Callbacks

- Your-Plugin/
 - html/
 - Callbacks/
 - Your-Plugin/
 - Elements/Login/
 - BeforeForm

HTML::Mason Callbacks

```
<div style="color: red; width: 30em; margin-right: auto; margin-left: auto">  
Please remember to use your Fooid usernames  
</div>
```

HTML::Mason Callbacks

Please remember to use your official usernames

Login 3.8.HEAD

Username:

Password:

HTML::Mason Overlays

- `share/html/Elements/Login`
- `local/html/Elements/Login`
- hack it up as needed
 - You lose on upgrades, especially for large elements
 - Patches for refactoring or callbacks generally welcome

perl Overlays

- `local/lib/RT/Ticket_Vendor.pm`
- `local/lib/RT/Ticket_Local.pm`
- `lib/RT/Sample/YAPC2010.pm`

perl Overlays

- `local/plugins/RT-Authen-ExternalAuth/lib/RT/User_Overlay.pm`
- `local/plugins/RT-Extension-MergeUsers/lib/RT/User_Overlay.pm`

perl Overlays

- `lib/RT/Sample/YAPC2010.pm`

perl Overlays

```
package RT::Interface::Web;
no warnings 'redefine';
sub WebCanonicalizeInfo {
    my $user = $ENV{REMOTE_USER};
    $user =~ s/\@FOOCorp\.com//i;
    return $user;
}
```

perl Overlays

- please don't copy the whole file
- you only need the method you want to clobber
- please don't copy the whole file
- the more code you copy, the worse your upgrade will be

Building an RT Extension

Development RT

```
git clone git://github.com/bestpractical/rt.git
```

```
git checkout rt-3.8.8 (or work on 3.8-trunk)
```

```
./configure --enable-layout=inplace --with-db-type=SQLite
```

```
make initdb
```

```
Set($MailCommand, 'testfile');
```

```
rm -rf var/mason/obj/* && ./bin/standalone_httpd 8012
```

RT Extension Pieces

- Module::Install::RTx
 - Makefile.PL
 - etc/initialdata
- lib/RT/Sample/YAPC2010.pm
- html/Callbacks/YAPC
 - html/Callbacks/RT-Sample-YAPC2010
- html/Overlays

Module::Install::RTx

- Make sure you're on 0.25
- Make sure you're on RT > 3.8.2
 - (preferably 3.8.6 for security)

Module::Install::RTx

- name 'RT::Sample::YAPC2010';
- RTx 'RT::Sample::YAPC2010';
- perl Makefile.PL
 - RTHOME=/path/to/testrt perl Makefile.PL

Using the new Plugin

- Make sure you have a `lib/RT/PluginName.pm`
- `lib/RT/Sample/YAPC2010.pm`
- `return I;`
- `make install`
- `Set(@Plugins,qw(RT::Sample::YAPC2010));`

Making the plugin do stuff

- `html/`
 - `Callbacks/`
 - `RT-Sample-YAPC2010/`
 - `Elements/Login/`
 - `BeforeForm`

Callbacks

- add in the warning code
- make install
- restart apache
 - or use dev mode

CSS Callbacks

- `html/Callbacks/Plugin-Name/`
 - `NoAuth/css/web2/main.css/Begin`
 - `NoAuth/css/web2/main.css/End`

Overlays

- RT-Sample-YAPC2010
 - html/Elements/Login
 - lib/RT/User_Local.pm
 - lib/RT/Sample/YAPC2010.pm

initialdata

- `rt-3.8.8/etc/initialdata` for samples
- `lib/RT/Handle.pm` for the implementation

initialdata

```
@CustomFields = (  
  {  
    Name      => 'Department',  
    Type      => 'FreeformSingle',  
    Queue     => 0,  
    Disabled  => 0,  
  },  
  {  
    Name      => 'Building',  
    Type      => 'FreeformSingle',  
    Queue     => [qw(General Sales)],  
    Disabled  => 0,  
  },  
);
```

initialdata

```
{  
  Name      => 'Billable',  
  Type      => 'SelectSingle',  
  Queue     => 0,  
  Disabled  => 0,  
  Values    => [  
    { Name   => "yes", SortOrder => 1 },  
    { Name   => "no",  SortOrder => 2 },  
  ],  
}
```

initialdata

```
@Users =  
  map {Name => $_, EmailAddress => "$_ \@foo", Privileged => 1 },  
  qw/Kevin Bob User2/;
```

initialdata

```
@ScripActions = (  
  {  
    Name      => 'Set default CF values from requestor',  
    ExecModule => 'DefaultCustomFields',  
  },  
);  
  
@ScripConditions = (  
  {  
    Name          => 'Department Change',  
    ApplicableTransTypes => 'Any',  
    ExecModule    => 'DepartmentChange'  
  }  
);  
  
@Scrips = (  
  {  
    Description  => 'Set values from Requestor',  
    ScripCondition => 'On Create',  
    ScripAction  => 'Set default CF values from requestor',  
    Template     => 'Blank',  
  },  
);
```

initialdata

```
@ScripActions = (  
  {  
    Name      => 'Set default CF values from requestor',  
    ExecModule => 'DefaultCustomFields',  
  },  
);  
  
@ScripConditions = (  
  {  
    Name          => 'Department Change',  
    ApplicableTransTypes => 'Any',  
    ExecModule     => 'DepartmentChange'  
  }  
);  
  
@Scrips = (  
  {  
    Description   => 'Set values from Requestor',  
    ScripCondition => 'On Create',  
    ScripAction   => 'Set default CF values from requestor',  
    Template      => 'Blank',  
  },  
);
```

initialdata

```
@ScripActions = (  
  {  
    Name      => 'Set default CF values from requestor',  
    ExecModule => 'DefaultCustomFields',  
  },  
);  
  
@ScripConditions = (  
  {  
    Name          => 'Department Change',  
    ApplicableTransTypes => 'Any',  
    ExecModule    => 'DepartmentChange'  
  }  
);  
  
@Scrips = (  
  {  
    Description   => 'Set values from Requestor',  
    ScripCondition => 'On Create',  
    ScripAction   => 'Set default CF values from requestor',  
    Template      => 'Blank',  
  },  
);
```

initialdata

- @Groups
- @Users
- @ACL
- @Queues
- @ScripActions
- @ScripConditions
- @Templates
- @CustomFields
- @Scrips
- @Attributes
- @Initial
- @Final

Things in the RT code that should confuse you

```
my $Ticket = RT::Ticket->new($RT::SystemUser);
```

```
my $Ticket = RT::Ticket->new($session{CurrentUser});
```

Things in the RT code that should confuse you

`$RT::Nobody`

`$RT::SystemUser`

`$Ticket->CurrentUser`

`$Ticket->CreatorObj`

Things in the RT code that should confuse you

- DBIx::SearchBuilder
- DBIx::SearchBuilder::Record
 - Limit(FIELD => ..., VALUE =>)
 - Next
 - BuildSelectQuery

Things in the RT code that should confuse you

- Principals
 - Groups
 - Users
- `$User->PrincipalObject()`
 - I'm sorry this isn't more consistent

Things in the RT code that should confuse you

- `loc("foo")`
- `loc("[_]'s first",$foo)`
- `<&|/|&>Foo</&>`

**And now a random
collection of useful bits**

patching RT

```
git clone git://github.com/bestpractical/rt.git
```

```
git checkout -b localhacking
```

```
git add $files
```

```
git ci -m 'a great commit message'
```

```
git format-patch 3.8-trunk
```

rt.cpan.org modules

- <http://github.com/bestpractical/cpan2rt>
- http://github.com/bestpractical/rt-extension-rt_cpan_org
- <http://github.com/bestpractical/rt-bugtracker-public>
- <http://github.com/bestpractical/rt-bugtracker>

rt.cpan.org modules

- RT::Authen::(Bitcard|OpenID|PAUSE)
- RT::Extension::ReportSpam
- RT::Extension::QuickDelete
- RT::Extension::MergeUsers

rt.cpan.org modules

Port a bugfix from rt-bugtracker

Allow people to search by Dist::Name instead of Dist-Name



jibsheet (author)

11 minutes ago

```
commit b669929d8a0c8b183454
tree 2ee0c3dbe490b924356f
parent fd314a1858c3131e6ed4
```

html/Public/index.html

1

html/Public/index.html

[View file @ b669929](#)

```
...  ... @@ -108,6 +108,7 @@ if ( defined $Maintainer && length $Maintainer ) {
108 108
109 109   if ( defined $Distribution && length $Distribution ) {
110 110     my $queue = RT::Queue->new( $session{'CurrentUser'} );
111 +   $Distribution =~ s/::/-/g;
111 112     $queue->Load( $Distribution );
112 113     unless ( $queue->id ) {
113 114       push @results, loc("No distribution '[_1]' in the system", $Distribution);
```

rt.cpan.org modules

- When you complain, I'll probably point at github for simple UI tweaks like this

rt.cpan.org Modules

Logged in as FALCONE | [Preferences](#) | [About rt.cpan.org](#) | [Logout](#)

[New ticket in](#) [Search...](#)

Please report any issues with rt.cpan.org to rt-cpan-admin@bestpractical.com.

[Edit](#)

Config Options

- `Set($YourConfig, 'foo');`
- `RT->Config->Get('YourConfig')`
- `RT->Config->Get('YourConfig',$CurrentUser)`

Config Options

```
SearchResultsRefreshInterval => {  
  Section      => 'General',          #loc  
  Overridable  => 1,  
  SortOrder    => 8,  
  Widget       => '/Widgets/Form/Select',  
  WidgetArguments => {  
    Description => 'Search results refresh interval',          #loc  
    Values      => [qw(0 120 300 600 1200 3600 7200)],  
    ValuesLabel => {  
      0 => "Don't refresh search results.",          #loc  
      120 => "Refresh search results every 2 minutes.", #loc  
      300 => "Refresh search results every 5 minutes.", #loc  
      600 => "Refresh search results every 10 minutes.", #loc  
      1200 => "Refresh search results every 20 minutes.", #loc  
      3600 => "Refresh search results every 60 minutes.", #loc  
      7200 => "Refresh search results every 120 minutes.", #loc  
    },  
  },  
},
```

Config Options

```
# General user overridable options
DefaultQueue => {
  Section      => 'General',
  Overridable  => 1,
  SortOrder    => 1,
  Widget       => '/Widgets/Form/Select',
  WidgetArguments => {
    Description => 'Default queue', #loc
    Callback   => sub {
      my $ret = { Values => [], ValuesLabel => {} };
      my $q = RT::Queues->new($HTML::Mason::Commands::session{'CurrentUser'});
      $q->UnLimit;
      while (my $queue = $q->Next) {
        next unless $queue->CurrentUserHasRight("CreateTicket");
        push @{$ret->{Values}}, $queue->Id;
        $ret->{ValuesLabel}{$queue->Id} = $queue->Name;
      }
      return $ret;
    },
  },
},
```

Config Options

```
$RT::Config::META{YourConfig} =  
  {  
    Section      => 'General',  
    Overridable  => 1,  
    SortOrder    => 50,  
    Widget       => '/Widgets/Form/Boolean',  
    WidgetArguments => {  
      Description => 'Do it! or not.', #loc  
    },  
  };
```

Config Options

General

Default queue

Username format

Theme

WYSIWYG message composer Yes No Use default (Yes)

WYSIWYG composer height Default: 200

Message box width Default: 72

Message box height Default: 15

Search results refresh interval

Default Update Type when Resolving

Do it! or not. Yes No Use default (No)

Custom Conditions

^ **User Defined conditions and actions**

(Use these fields when you choose 'User Defined' for a condition or action)

Custom condition:

Custom action preparation code:

Custom action cleanup code:

Custom Conditions

```
package RT::Condition::DepartmentChange;
use base qw(RT::Condition);

sub IsApplicable {
    my $self = shift;
    my $T = $self->TransactionObj;
    my $Department = RT::CustomField->new($RT::SystemUser);
    my ($ok, $msg) = $Department->Load('Department');
    unless ($ok) {
        $RT::Logger->error("Failed to load Department CF: $msg");
        return 0;
    }

    if ( $T->Type eq 'Create' ||
        ( $T->Type eq 'CustomField' && $T->Field == $Department->Id ) ) {
        return 1;
    }

    return 0;
}

1;
```

Custom Actions

^ **User Defined conditions and actions**

(Use these fields when you choose 'User Defined' for a condition or action)

Custom condition:

Custom action preparation code:

Custom action cleanup code:

Custom Actions

```
package RT::Action::DefaultCustomFields;
use base qw(RT::Action);

sub Prepare { I };

sub Commit {
    my $self = shift;
    my $T = $self->TicketObj;
    $T->AddCustomFieldValue(Field => 'Billable', Value => 'no')
        unless $T->CheckSomething;
};
```

Extension Images

- you can't use `html/NoAuth/images`
 - autohandler--
 - remove

`Alias /NoAuth/images/ /opt/rt3/share/html/NoAuth/images/`

Extension Images

```
<%INIT>
use File::Basename;
my $arg = $m->dhandler_arg;
my $file = dirname($m->current_comp->source_file) . '/real/' . $arg;
return $m->decline unless -f $file && -r _;

my $suffix = $1 if $file =~ /([\^.]+)$/;

my $type = {qw(
    xml      text/xml
    html     text/html
    js       application/javascript
    css      text/css
    gif      image/gif
    jpg      image/jpeg
    png      image/png
)}->{$suffix} || 'application/octet-stream';

RT::Interface::Web->SendStaticFile( File => $file, Type => $type );

$m->abort;
</%INIT>
```

Questions

- [git://github.com/jibsheets/rt-sample-yapc2010](https://github.com/jibsheets/rt-sample-yapc2010)
- <http://jibsheets.com/talks/yapc2010.pdf>
- [#rt on irc.perl.org](https://irc.perl.org/#rt)
- [\(rt-users|rt-devel\)@lists.bestpractical.com](mailto:(rt-users|rt-devel)@lists.bestpractical.com)